Generating Features for Named Entity Recognition by Learning Prototypes in Semantic Space: The Case of De-Identifying Health Records

Aron Henriksson
Department of Computer and Systems Sciences (DSV)
Stockholm University
Stockholm, Sweden
Email: aronhen@dsv.su.se

Hercules Dalianis
Department of Computer and Systems Sciences (DSV)
Stockholm University
Stockholm, Sweden
Email: hercules@dsv.su.se

Stewart Kowalski
Department of Computer and Systems Sciences (DSV)
Stockholm University
Stockholm, Sweden
Email: stewart@dsv.su.se

***Abstract*—Creating sufficiently large annotated resources for supervised machine learning, and doing so for every problem and every domain, is prohibitively expensive. Techniques that leverage large amounts of unlabeled data, which are often readily available, may decrease the amount of data that needs to be annotated to obtain a certain level of performance, as well as improve performance when large annotated resources are indeed available. Here, the development of one such method is presented, where semantic features are generated by exploiting the available annotations to learn prototypical (vector) representations of each named entity class in semantic space, constructed by employing a model of distributional semantics (random indexing) over a large, unannotated, in-domain corpus. Binary features that describe whether a given word belongs to a specific named entity class are provided to the learning algorithm; the feature values are determined by calculating the (cosine) distance in semantic space to each of the learned prototype vectors and ascertaining whether they are below or above a given threshold, set to optimize F_β-score. The proposed method is evaluated empirically in a series of experiments, where the case is health-record deidentification, a task that involves identifying protected health information (PHI) in text. It is shown that a conditional random fields model with access to the generated semantic features, in addition to a set of orthographic and syntactic features, significantly outperforms, in terms of F_1-score, a baseline model without access to the semantic features. Moreover, the quality of the features is further improved by employing a number of slightly different models of distributional semantics in an ensemble. Finally, the way in which the features are generated allows one to optimize them for various F_β-scores, giving some degree of control to trade off precision and recall. Methods that are able to improve performance on named entity recognition tasks by exploiting large amounts of unlabeled data may substantially reduce costs involved in creating annotated resources for every domain and every problem.**

I. INTRODUCTION

The ability to recognize, in text, references to entities of certain semantic categories – a task known as named entity recognition (NER) – is fundamental for information extraction. In recent years, information extraction in the medical domain has proliferated [1], not least with the ongoing efforts to facilitate information reuse of clinical data [2], which are recorded daily on a massive scale in electronic health records (EHRs). Since the majority of these data are in the form of text, natural language processing (NLP) methods need to be adapted and applied to this rather peculiar text genre, or sublanguage [3] – often not complying with formal grammar and littered with misspellings and non-standard shorthand – in order to enable effective exploitation of this data source in the ultimate endeavor to improve health care. An example of a medical NER system is one that is able to recognize mentions of, for instance, symptoms, disorders and drugs in EHRs. Such a system can then form an essential building block of various medically useful applications: generating patient problem lists [4], comorbidity analyses [5], syndromic surveillance [6] and adverse drug event detection [7]–[9].

The most common approaches to NER rely on dictionaries, hand-crafted rules or machine learning, or some combination of these [10]. Relying solely on dictionaries is often insufficient, as they tend to lack in coverage, and string-matching methods typically do not perform well with noisy data. Rules-based approaches, although they sometimes perform well, are costly and time-consuming to create and often are not generalizable to new datasets. As a result, statistical and machine learning approaches have become increasingly popular. The machine learning approach is typically supervised, which means that the learning algorithm uses labeled examples to construct a model. This latter approach hence relies on human annotations of named entities. Based on these, various types of features can be used: orthographic, syntactic and semantic. Manually creating annotated resources is, however, costly, and doing so for every (sub-)domain and task is cumbersome and prohibitively expensive. While entirely unsupervised approaches currently do not seem feasible, it would, however, be advantageous if the annotation effort could be limited as much as possible. One line of research that has emerged to tackle this problem aims to provide additional features by using unsupervised methods.

In this paper, we present an approach that uses models of distributional semantics in conjunction with a large, unannotated, in-domain corpus to provide additional features to the learning algorithm. It is shown that using these features in conjunction with a set of baseline features – orthographic and syntactic – significantly outperforms, in terms of F_1-

978-1-4799-5669-2/14/$31.00 ©2014 IEEE

score, using only the baseline features. In contrast to similar approaches, this method exploits the existing (named entity) annotations to create *prototype vectors* in semantic space that are intended to capture the meaning of a particular semantic category. Moreover, rather than using a single model of distributional semantics, which is the traditional approach, we experiment with combining multiple models, hypothesizing that combining multiple semantic spaces will lead to the creation of features that more holistically capture the meaning of a semantic category.

The proposed feature-generating method is here evaluated in the context of EHR deidentification, which can be, and often is, approached as a NER task, where certain sensitive information is to be identified and either removed or replaced. Deidentification of EHRs is an important area of research that aims to ensure the privacy of patients, while allowing clinical data to be used for research, as this is currently an underutilized resource, which, if unlocked, may lead to improved health care.

A. De-Identifying Health Records

While ensuring the security and privacy of EHR systems is important in itself [11], and is in most countries regulated by law, it also affects the possibility of using the data they contain for research purposes. In the United States, the Health Insurance Portability and Accountability Act (HIPAA) protects the confidentiality of patient data [12]. In order to use such data for research, informed consent of the patient and approval of the Internal Review Board typically have to be obtained. This requirement can, however, be waived if the data has been deidentified. Clinical data is considered deidentified, according to the HIPAA Safe Harbor technique, if 18 types of protected health information (PHI) have been removed. Deidentification[1], or scrubbing, of clinical text is often done manually, but as this requires significant resources, there have been many efforts in the research community to create automatic deidentification methods (see [13] for an overview). Important early steps to this end were taken in the Informatics for Integrating Biology to the Bedside (i2b2) deidentification challenge in 2006, in which seven teams participated [14].

Systems used for automatic deidentification of text-based health records can crudely be divided into (1) those that are based on (hand-crafted) rules and pattern-matching techniques, often in conjunction with dictionaries, and (2) those that are based on machine learning [13] – as well as combinations of the two [15]. The pros and cons of these two approaches are the same as those described earlier for other NER tasks. In general, pattern-matching methods that use dictionaries of, e.g., proper names, geographical locations, health care institutions tend to perform better for rare PHI, whereas machine learning systems tend to perform better with PHI that is not covered by the employed dictionaries [13]. In the i2b2 deidentification challenge, systems based on machine learning performed best [14]. Most systems based on machine learning used conditional random fields or support vector machines to train their models, using various types of features: lexical (typically attributes of tokens: casing, punctuation, morphology, etc.), syntactic (typically part-of-speech) and semantic (typically semantic classification of tokens according to dictionaries or semantic types) [13]. To the best of our knowledge, however, the use of distributional semantic features for deidentification has not been studied previously.

[1]Note that deidentification is not the same as anonymization, which requires that the patient cannot be reidentified from the data; deidentification requires only that explicit identifiers are removed [13].

B. Generating Features with Distributional Semantics

The idea to improve the predictive performance of a supervised NLP system by providing the learning algorithm with unsupervised word representations as additional features has been around for a while. The motivation behind this is that sparsity in the labeled training data can be reduced by using unlabelled data, effectively improving the generalization accuracy of semi-supervised approaches. A popular method for constructing these word representations is to use clustering – see, for instance, [16]. Word representations can, however, be induced in many other ways: several of these are compared in [17], where it is shown that, by adding semantic word representations, near state-of-the-art supervised baselines for both NER and chunking can be outperformed.

Some of these can be categorized as models of distributional semantics, which exploit the distributional structure of language to acquire the meaning of words. Distributional semantics is a topic that has received much attention in the NLP research community and has lately been explored in the biomedical [18] and clinical [19] domains. One of the few studies to improve NER on clinical text is described in [20].

II. METHODS AND MATERIALS

The method presented here presupposes the existence of two resources: (1) an annotated (named entity) corpus and (2) a much larger unannotated corpus, preferably in the same domain. The method essentially comprises the following steps:

1) Learning prototypical representations of the target classes in semantic space
2) Generating features for the instances (words) based on their distance to each of the prototype vectors
3) Applying an appropriate learning algorithm to the annotated corpus with the generated features

The focus of this paper is primarily on the first two steps, which are essentially about providing additional features, with the use of distributional semantics, to the learning algorithm. A number of ways of providing features are explored and compared, including the use of multiple distributional semantic models. Ultimately, the method is evaluated in the third step, where a predictive model with access to the additional features is compared to a predictive model that only has access to a set of baseline features.

A. Learning Prototype Vectors

An abstract representation of each target (named entity) class is created by making use of the existing annotations in conjunction with a semantic space induced from a large, unannotated corpus. Here, this is achieved by taking the centroid of the semantic vectors that represent the instances – which, in this case, are words – that have been annotated as belonging to a particular class and occur at least 100 times in the unannotated corpus. This results in an abstract representation, i.e., one that does not correspond to an actual

instance, as is otherwise often the case when calculating the centroid of a cluster. The centroid is defined as the median value of each dimension (see Algorithm 1).

input : Set V of n-dimensional vectors $\{\overrightarrow{\mathbf{v}}\}$
output: n-dimensional prototype vector $\overrightarrow{\mathbf{p}}$

for $i \leftarrow 1$ **to** n **do**
  for $\overrightarrow{\mathbf{v}} \in V$ **do**
    $coordinates \leftarrow \overrightarrow{\mathbf{v}}_i$
  end
  $\overrightarrow{\mathbf{p_i}} \leftarrow median(coordinates)$
end
return $\overrightarrow{\mathbf{p}}$

Algorithm 1: Learning Prototype Vectors

Here, three different yet related distributional semantic models, all based on random indexing [21], are used to build the semantic spaces. Random indexing is a scalable and computationally efficient method for creating semantic spaces. It creates a reduced-dimensional space in which the relative distances between vectors have been approximately preserved. In contrast to other dimensionality reduction techniques like singular value decomposition (SVD) – as well as the distributional semantic models that rely on SVD, e.g., latent semantic analysis/indexing – it circumvents the need to construct a term-by-term matrix that is subsequently reduced. Instead, pre-reduced vectors[2] are incrementally populated with co-occurrence information.

In the construction of a semantic space with random indexing, there are two types of vectors: *index vectors*, which are used only in the construction phase, and *semantic vectors*, which represent the meaning of words and collectively make up the actual semantic space. Each unique word w_j in our vocabulary W is assigned an index vector $\vec{w}_j^i$ and a semantic vector $\vec{w}_j^s$ of dimensionality d, which in our case is set to 5,000. The index vectors are static representations of the words that are approximately uncorrelated to each other. This is achieved by creating very sparse vectors that are randomly assigned a small number of non-zero elements (1s and -1s), in our case fifty (1%), with equally many 1s and -1s. A $\vec{w}_j^s$ – containing the distributional profile of the word w_j – is then the sum of all the index vectors of the words with which w_j co-occurs within a window of a certain size. In these experiments, a window size of two words to the left and two words to the right of the target word was used, as this has been shown to capture both synonymy [22] and wider semantic categories [23] well.

In this setting, word order within the context window is effectively ignored; however, it is also possible to encode word order information in the semantic vectors by permuting the elements of the index vectors on the fly before adding them to $\vec{w}_j^s$ [24] – this has been shown to improve performance on various synonym extraction task [24], [25]. The semantic vectors are referred to as **order vectors** when the elements in the index vectors are shifted according to their corresponding words' relative position to the target word: for a word that occurs two positions to the left of the target word, the elements of that word's index vector are shifted two positions to the left before adding the index vector to the semantic vector, and for a word that occurs one position to the right of the target word, the elements are shifted once to the right. The semantic vectors are referred to as **direction vectors** when the elements in the index vectors are shifted only one position depending on whether the corresponding word occurs to the left or the right of the target word. When the element vectors are not shifted at all, the semantic vectors are sometimes referred to as **context vectors**.

[2]The vectors are pre-reduced in the sense that their dimensionality is configured to be much smaller than the size of the vocabulary.

B. Generating Distributional Features

Once we have prototype vectors for each of our target classes, we need a means of using these to generate features for all the instances in our dataset, both in our training set and in our test set. In this study we have one feature per named entity class (and distributional model), where the value is either True or False depending on whether the cosine similarity between the target word and the prototype vector is above a set threshold. The threshold is based on the distances between the annotated named entities and its corresponding prototype vector. The threshold is selected with the optimization objective to maximize F_β-score (Eq. 1).

$$\arg\max_{t \in V} \left((1+\beta^2) \frac{P^{(t)} \cdot R^{(t)}}{(\beta^2 \cdot P^{(t)}) + R^{(t)}} \right), \qquad (1)$$

where P is precision (true positives / true positives + false positives) and R is recall (true positives / true positives + false negatives); $V = (0, 0.0001, 0.0002, ..., 1)$; β determines the weight that should be given to recall relative to precision. The lowest threshold is chosen that optimizes the F_β-score.

C. Training Named Entity Recognition Model

In addition to the generated distributional semantic features, a set of orthographic and syntactic features are also generated. These features are commonly used for NER and are the same as the ones used in [26]. In this paper, we refer to these thirteen feature as *baseline features*:

F_1:	Is the token alphanumeric?
F_2:	Is the token numeric?
F_3:	Does the token have an initial capital letter?
F_4:	What is the part-of-speech tag of the token?
F_{5-8}:	What is the part-of-speech tag of one/two token(s) before/after the current token?
F_9:	What is the length of the token?
F_{10-13}:	What is the token length of the token(s) one/two before/after the current token?

These features, in addition to the generated semantic features, are then provided to the learning algorithm together with the class label, which contains the PHI class of the current token. Here, the considered learning algorithm is conditional random fields (CRF) as implemented in CRF++[3].

Conditional random fields (see [27] for an introduction) is a popular choice for structured prediction tasks, i.e., when

[3]http://code.google.com/p/crfpp/

sequential data needs to be segmented and/or labeled, which NER is an example of. The power of CRF lies in its ability to model multiple variables that are dependent on each other – as they typically are in structured prediction tasks – and to exploit large sets of input features. It achieves this by using an undirected probabilistic graphical model that, unlike, e.g., Hidden Markov Models (which is generative), is discriminative. Here, we use a linear-chain CRF that, in addition to being dependent on the input features, is also dependent on the previous and subsequent output variables. In the subsequent experiments, CRF model parameters are not optimized; instead, the same parameters settings as in [26] are used: a C-value[4] of 5, a context window of 2+2 and using L2-regularization.

D. Data and Experimental Setup

The two corpora that are used in this study are subsets of the Stockholm EPR Corpus[5] [28], [29], which comprises health records from a wide range of health care units at Karolinska University Hospital in Stockholm, Sweden over a five-year period (2006-2010). The two corpora are: (1) a small annotated PHI corpus and (2) a large unannotated corpus. The Stockholm EPR PHI Corpus [30] comprises one hundred health records from five different clinics (Neurology, Orthopedics, Infection, Dental Surgery, and Nutrition). This corpus originally contained 28 PHI classes that were annotated by three annotators – see [31] for a detailed description of the corpus creation process. A consensus-based gold standard was later derived from the original annotations after discussions between the annotators [30]. This process included merging conceptually similar PHI classes, resulting in the following eight classes[6], along with a *Non-PHI* class: *First Name*, *Last Name*, *Age*, *Health Care Unit*, *Location*, *Full Data*, *Date Part* and *Phone Number*. The version of the PHI corpus used in the following experiments contains a total of 198,821 instances, the vast majority of which belongs to the *Non-PHI* class. The unannotated corpus, which is used for building the semantic spaces, with which additional features are derived, contains ten million clinical notes and approximately 169 million tokens (2.3 million types). Three semantic spaces are constructed with three distributional semantic models: context vectors, direction vectors and order vectors.

In the first experiment, two ways of obtaining prototype vectors for the PHI classes are compared, where the centroid vector is defined as either the mean or the median, and the hypothesis is that it will be easier to separate the classes with the use of median vectors compared to mean vectors since these will be less sensitive to outliers. This experiment is conducted using only the training data, where the F_1-score for each PHI class is calculated using the obtained optimal cosine similarity thresholds. The results are macro-averaged over PHI classes and the three distributional semantic models.

In the second experiment, the quality of the distributional features, in terms of their positive impact on the NER results, produced by three distributional semantic models – context vectors, direction vectors and order vectors – is compared.

In the third experiment, different ways of combining the three distributional semantic models are evaluated and compared. In F: $\mathrm{M}_1 \wedge\ \mathrm{M}_2 \wedge\ \mathrm{M}_3$, the final feature value is True only if it is True in all single models; in F: $\mathrm{M}_1 \vee\ \mathrm{M}_2 \vee\ \mathrm{M}_3$, the final feature value is True if it is True in at least one of the single models; in F: Majority, the final feature value is True if it is True in at least two of the three single models. All of the above combination strategies result in an additional eight features, one per PHI class. In F: $\mathrm{M}_1 +\ \mathrm{M}_2 +\ \mathrm{M}_3$, the feature values produced by each of the single models are all used, resulting in 24 additional features.

In the fourth experiment, which is the central one in this study, three CRF models are compared: (1) using a set of baseline features, (2) using the baseline features along with features generated by a single distributional model, and (3) using the baseline features along with features generated by a combination of three distributional models.

In the fifth experiment, the optimization objective in the feature generation procedure is varied to F_β-scores with different β values. The purpose of this experiment is to investigate the impact of the precision and recall of the generated features on the precision, recall and F_1-score of the NER task. The following β values are used: 0.5, 1.0, 2.0 and 5.0.

In terms of evaluation, the considered performance metrics are precision, recall and F_1-score[7]. In all experiments, 10-fold cross-validation is carried out. Performance scores are macro-averaged over classes, giving equal weight to all classes. In the first experiment, where two competing methods are compared, the Wilcoxon signed-rank test is employed for statistical hypothesis testing, where the null hypothesis is that the methods perform equally well. This test ranks the differences in performance of two feature representations on each dataset, ignoring the signs, and compares the ranks of positive and negative differences. It was chosen for its robustness when comparing two classifiers [32]. In the fourth experiment, two types of significance tests are conducted. In the first, which involves comparisons of multiple methods, a Friedman test is employed, followed by a post-hoc test using the Bergman-Hommel procedure, as suggested in [33]. Again, the ranks are compared, but now adjusting for the fact that multiple comparisons are performed. In the second, where pairwise differences are compared for specific classes, i.e., on an instance level, McNemar's test with continuity correction is conducted [34].

III. RESULTS

The series of experiments that was conducted in this study can be divided into five parts: (1) comparing two ways of learning prototype vectors, (2) comparing the impact of using features derived from different models of distributional semantics, (3) investigating a number of strategies for combining features derived from multiple semantic spaces, (4) studying the impact of using various feature sets on the targeted NER task, and (5) varying the optimization objective in the

[4]This hyper-parameter trades the balance between overfitting and underfitting: with a higher value, the risk of overfitting the training data increases.

[5]This research has been approved by the Regional Ethical Review Board in Stockholm (Etikprövningsnämnden i Stockholm), permission number 2012/834-31/5.

[6]*E-mail Address* and *Social Security Number* were rare in the annotated corpus and were therefore removed.

[7]Precision corresponds to positive predictive value and recall corresponds to sensitivity. F_1-score is the harmonic mean of precision and recall.

prototype learning phase. The results of the experiments are described below according to this division.

A. Prototype Vectors: Mean vs. Median Centroids

In the prototype learning phase, where the prototype vector of each PHI class is defined as the centroid of the annotated instances' context vectors, two ways of constructing the prototype vector are compared: taking the mean or median of the values in each dimension. The F_1-scores obtained using the optimal thresholds on the training data, averaged over folds and the three distributional models, are reported in Table I. Defining the centroid as the median vector rather than the mean vector results in a higher F_1-score for all eight PHI classes – often with a very large margin – and the Wilcoxon signed rank test shows that this difference is significant ($p < 0.01$) across classes. The F_1-score, macro-averaged over the PHI classes, is much higher when using median vectors (0.36) compared to using mean vectors (0.13). In all subsequent experiments, the centroid is hence defined as the median vector.

TABLE I. AVERAGE F_1-SCORES WITH CENTROIDS AS MEAN OR MEDIAN VECTORS WHEN SETTING THRESHOLDS USING TRAINING FOLDS

Class	Mean Vector	Median Vector
First Name	0.03666	**0.40745**
Last Name	0.40781	**0.44191**
Age	0.07022	**0.51707**
Health Care Unit	0.00845	**0.30829**
Location	0.00111	**0.33264**
Full Date	0.07362	**0.31118**
Date Part	0.36312	**0.42453**
Phone Number	0.13719	**0.14584**
Macro Average	0.12565	0.36111
P-value	0.007813	

B. Comparing Models of Distributional Semantics

The impact on the NER task of providing the CRF learning algorithm with semantic features derived from three different models of distributional semantics was then analyzed. The results, summarized in Table II, show that using direction vectors results in a higher macro-averaged precision, recall and F_1-score, compared to using context vectors or order vectors.

TABLE II. NER PERFORMANCE WHEN ADDING FEATURES DERIVED FROM A SINGLE DISTRIBUTIONAL MODEL

Distributional Model	Precision	Recall	F_1-Score
Context Vectors	91.399	80.676	84.997
Direction Vectors	**92.045**	**80.854**	**85.243**
Order Vectors	92.033	80.810	85.202

This indicates that the distributional models generate different feature values. Each distributional model generates different prototype vectors and thresholds, as can be seen in Figure 1, which shows the threshold setting procedure for the three distributional models. It can be seen that the thresholds are generally lower and the F_1-scores higher for direction vectors and order vectors. The quality of the prototype vectors seems to reflected in their contribution to NER performance.

C. Combining Multiple Semantic Spaces

In an attempt to improve performance, semantic spaces induced from different models of distributional semantics are used to generate the semantic features; how to combine these features is explored in the following experiment, the results of which are presented in Table III. The best-performing combination strategy, in terms of recall and F_1-score is F: $M_1+ M_2+ M_3$, whereby the features generated by each distributional model are retained. F: $M_1\vee M_2\vee M_3$, however, results in the best precision; this is also the second-best performing combination strategy in terms of F_1-score.

TABLE III. NER PERFORMANCE WHEN ADDING FEATURES DERIVED FROM THREE DISTRIBUTIONAL MODELS IN THREE DIFFERENT WAYS

Combination Strategy	Precision	Recall	F_1-Score
F: $M_1\wedge M_2\wedge M_3$	91.939	80.556	85.094
F: $M_1\vee M_2\vee M_3$	**92.128**	81.004	85.440
F: Majority	92.031	80.976	85.371
F: $M_1+ M_2+ M_3$	91.786	**81.283**	**85.502**

D. Impact of Features on Named Entity Recognition

To study and verify the impact of providing various types of semantic features to the learning algorithm, the best setups from the previous experiments – the best-performing single distributional model (SDM) and multiple distributional models (MDM), respectively – are compared to a baseline model that only has access to a set of orthographic and syntactic features, but not semantic, features. The results, presented in Table IV, show that MDM obtains the highest macro-averaged recall and F_1-score; however, SDM obtains a slightly higher macro-averaged precision. A similar pattern is reflected in the average rankings, but in this case MDM and SDM have the same average ranking for precision. In terms of F_1-score, MDM is the best-performing model for five out of nine classes: it outperforms the baseline model for eight classes and SDM for six classes. Morever, MDM achieves a macro-averaged improvement of 1.83 percentage points over the baseline model, but only 0.26 percentage points over SDM. Comparing SDM to the baseline model yields a similarly clear trend: SDM achieves both higher macro-averaged scores and lower average rankings than the baseline model with all considered performance metrics.

A Friedman test shows that the differences in performance over classes between the three models are significant for F_1-score, but not for precision or recall. A post-hoc test shows that the differences in F_1-score between the baseline model and SDM (p-value = 0.03389), and the baseline model and MDM (p-value = 0.01403) are statistically significant; however, the difference between SDM and MDM is not (p-value = 0.47950).

Even if the differences in performance, in terms of precision and recall, are not significant overall, it is possible that differences are significant for certain classes. To ascertain whether that is the case, McNemar's test is conducted to compare the differences in precision and recall between the best-performing model with semantic features (MDM) and the baseline model. For precision, this results in significant differences for four classes: *First Name* ($p < 0.001$), *Last Name* ($p = 0.005$), *Age* ($p = 0.041$) and *Non-PHI* ($p < 0.001$). For recall, the number of significant differences amounts to six: *First Name* ($p < 0.001$), *Last Name* ($p = 0.039$), *Age* ($p = 0.041$), *Location* ($p = 0.007$), *Full Date* ($p = 0.018$) and *Date Part* ($p = 0.025$). For all of these, MDM outperforms the baseline model.

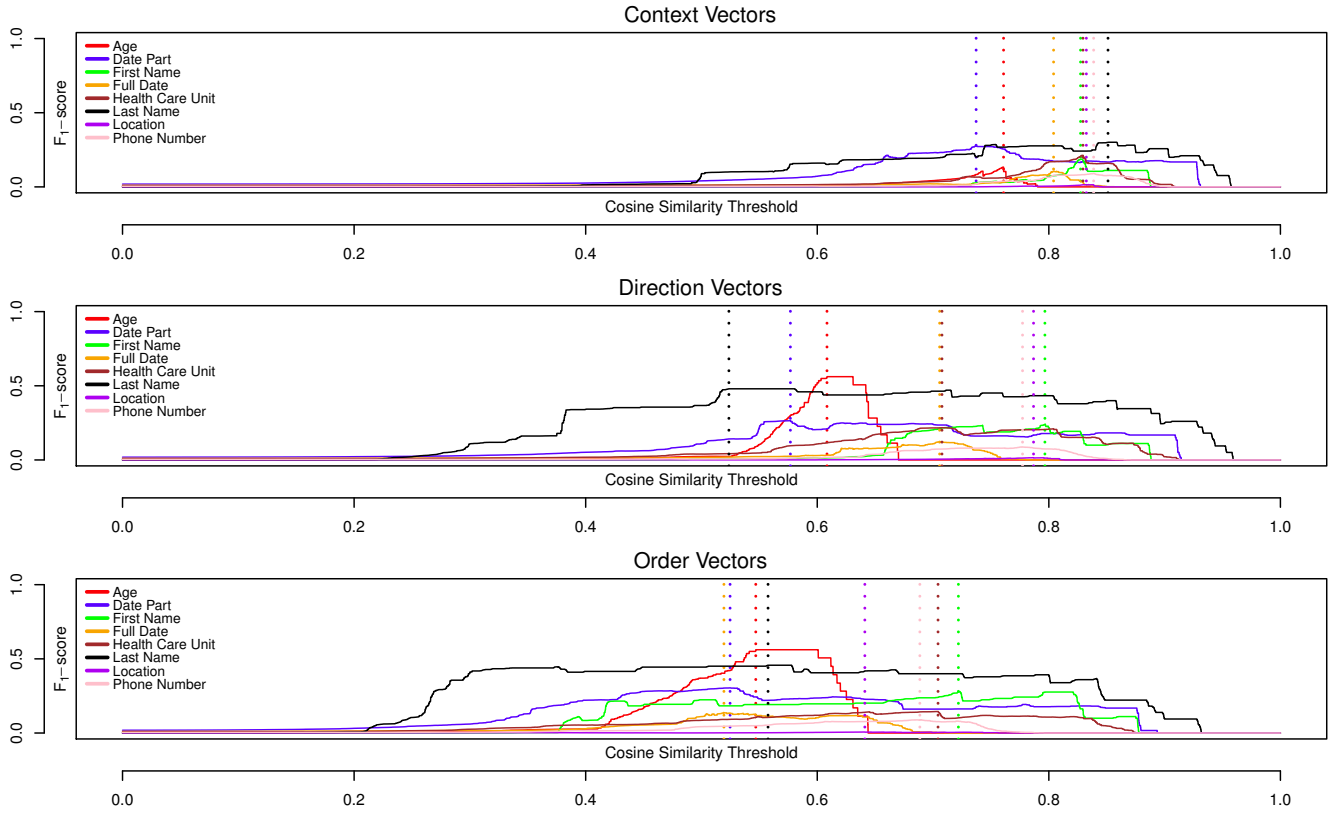

Fig. 1. Illustrating the setting of thresholds that maximize F_1-score for each PHI class, indicated by a dashed vertical line (these data are averaged over folds)

TABLE IV. NER PERFORMANCE SCORES AND RANKS PER PHI CLASS FOR CRF MODELS TRAINED WITH ORTHOGRAPHIC AND SYNTACTIC FEATURES (BASELINE), WITH ADDITIONAL SEMANTIC FEATURES DERIVED FROM A SINGLE DISTRIBUTIONAL MODEL (SDM) AND MULTIPLE DISTRIBUTIONAL MODELS (MDM). THE BEST RESULTS PER PHI CLASS ARE BOLDED; A † INDICATES A SIGNIFICANT DIFFERENCE ($p < 0.05$) BETWEEN MDM AND THE BASELINE.

		Precision (Rank)			Recall (Rank)			F_1-Score (Rank)		
Class	Instances	Baseline	SDM	MDM	Baseline	SDM	MDM	Baseline	SDM	MDM
First Name	803	93.239 (3)	94.588 (2)	**95.008 (1)**†	78.538 (3)	82.792 (2)	**83.290 (1)**†	85.160 (3)	88.179 (2)	**88.701 (1)**
Last Name	945	92.594 (3)	93.582 (2)	**94.382 (1)**†	87.148 (3)	88.570 (2)	**89.354 (1)**†	89.681 (3)	90.936 (2)	**91.704 (1)**
Age	85	94.666 (3)	**95.389 (1)**	95.250 (2)†	79.484 (3)	**85.317 (1)**	85.040 (2)†	84.840 (3)	**88.748 (1)**	88.726 (2)
Health Care Unit	1,322	**86.049 (1)**	84.616 (3)	84.785 (2)	62.755 (3)	**65.200 (1)**	64.265 (2)	72.267 (3)	**73.417 (1)**	73.084 (2)
Location	109	77.207 (2)	**80.341 (1)**	76.382 (3)	41.724 (3)	47.071 (2)	**48.040 (1)**†	53.373 (3)	57.857 (2)	**57.861 (1)**
Full Date	829	**93.437 (1)**	91.739 (3)	91.800 (2)	90.532 (2)	90.275 (3)	**91.942 (1)**†	**91.844 (1)**	90.853 (3)	91.728 (2)
Date Part	1,376	**91.727 (1)**	91.343 (2)	91.187 (3)	88.035 (3)	88.905 (2)	**89.036 (1)**†	89.800 (3)	**90.068 (1)**	90.059 (2)
Phone Number	316	93.197 (3)	**97.421 (1)**	97.197 (2)	**81.392 (1)**	79.761 (3)	80.786 (2)	86.492 (3)	87.544 (2)	**88.059 (1)**
Non-PHI	193,036	99.321 (3)	99.385 (2)	**99.388 (1)**†	**99.802 (1)**	99.796 (3)	99.798 (2)	99.560 (3)	99.589 (2)	**99.593 (1)**
Macro Average		91.271	92.045	91.786	78.823	80.854	81.283	83.669	85.243	85.502
Mean Rank		2.222	1.889	1.889	2.444	2.111	1.444	2.777	1.777	1.444
P-value			0.71653			0.09697			0.01312	

E. Varying Optimization Objective in Prototype Learning

To study the impact of changing the optimization objective to various F_β-scores, when generating the semantic features, on NER performance, experiments were conducted with the following β values: 0.5, 1.0, 2.0 and 5.0. The results of these experiments, presented in Table V, show how precision, recall and F_1-score – macro-averaged over classes – are affected by the optimization objective. The highest precision and F_1-score are observed when β is set to 2.0, and the highest recall is observed when β is set to 5.0.

TABLE V. THE IMPACT ON NER PERFORMANCE AS THE OPTIMIZATION OBJECTIVE OF THE FEATURE GENERATION IS VARIED

Optimization Objective	Precision	Recall	F_1-Score
F_β-score, $\beta = 0.5$	90.947	80.843	84.966
F_β-score, $\beta = 1.0$	91.786	81.283	85.502
F_β-score, $\beta = 2.0$	**92.005**	81.523	**85.702**
F_β-score, $\beta = 5.0$	91.001	**82.088**	85.678

IV. DISCUSSION

An important part of the feature-generating method presented in this paper concerns the use of prototype vectors in semantic space to determine whether a word belongs to a specific named entity class or not. This can be further broken down into two components: (1) learning prototype vectors and (2) setting thresholds that determine the feature values. To learn prototype vectors for each named entity class, the available annotations are exploited by calculating the centroid

of their distributional semantic representations. The hypothesis that median vectors would be better at robustly capturing the general meaning of a semantic category than mean vectors was shown in the first set of experiments to be valid. A plausible explanation for this, and the motivation behind the hypothesis, is that median vectors are less sensitive to outliers. Outliers may exist for several reasons: it could be due to, for instance, noise in the annotated corpus or cases of homonymy in the unannotated corpus, which may cause an annotated instance to have a conflicting representation in semantic space. The second component, in which thresholds are set for each named entity class by finding the cosine similarity that maximizes F_1-score on the training data, has the advantage that the optimization objective can be configured according to the priorities of a particular application. There may, however, be other methods that would do a better job of separating the classes in semantic space. One idea worth exploring in the future would be to use support vector machines to determine the feature values instead of setting thresholds in this manner.

The thresholds-setting procedure does, however, provide some interesting insights into differences between the three considered models of distributional semantics. The fact that thresholds are higher with context vectors compared to both direction vectors and order vectors is probably due to differences in density of the vectors: the shifting of (index) vector elements when constructing semantic spaces with the latter models means that the resulting vectors will be denser and, as a result, cosine similarity scores are more likely to be lower. Moreover, with these models, we are better able to separate the classes, indicating that taking into account word order is – to some degree – important to capture the meaning of a named entity class. This was confirmed in the subsequent experiments, the results of which showed that the highest observed NER performance was obtained with direction vectors, and the lowest with context vectors. It should be noted, however, that the macro-averaged scores are only marginally different and are unlikely to be significant across classes.

To improve the quality of the features, the three models of distributional semantics were then combined in various ways. Limiting ourselves to working with the output of each model, four strategies were devised: three of them can be said to put various demands on the precision versus recall of the feature values – with F: $\mathrm{M}_1\wedge\ \mathrm{M}_2\wedge\ \mathrm{M}_3$ prioritizing precision the most and F: $\mathrm{M}_1\vee\ \mathrm{M}_2\vee\ \mathrm{M}_3$ prioritizing recall the most – while the fourth simply provides the output of all three models to the learning algorithm. The experiment resulted in small differences in macro-averaged performance scores between the four combination strategies; however, F: $\mathrm{M}_1+\ \mathrm{M}_2+\ \mathrm{M}_3$ resulted in the highest observed recall and F_1-score. The fact that this, followed by F: $\mathrm{M}_1\vee\ \mathrm{M}_2\vee\ \mathrm{M}_3$, produced the best results seems to indicate that prioritizing recall in the feature-generating process is advantageous for the overall NER performance, also in terms of precision.

Arguably the most important experiment reported on in this paper is that in which the benefit of providing semantic features, in addition to the baseline – orthographic and syntactic – features to the learning algorithm is demonstrated, yielding better performance on the NER task. This is indeed shown, as there is a significant difference in terms of F_1-score between the three models – baseline, SDM and MDM – and that both SDM and MDM significantly outperform the baseline model across classes. Moreover, MDM produces significantly better precision and recall for a number of classes. The hypothesis that performance could be further improved by combining multiple semantic spaces could, however, not be validated in these experiments, although the observed performance did receive a boost. The fact that the F_1-score was higher for six classes with MDM compared to SDM does provide some indication that results may be improved by utilizing multiple distributional models. This – creating and employing ensembles of semantic spaces to improve performance on a variety of natural language processing tasks – is a line of research that has shown early promise [22] and one that we aim to continue to pursue.

The final experiment, in which various optimization objectives were used and where using F_2-score resulted in the overall highest F_1-score, again, indicates that over-generating – in the sense of prioritizing recall over precision – in the feature-generating procedure results in better NER performance. The ability to specify the optimization objective in terms of different F-scores is an advantage, as it gives some amount of control over the precision-recall trade-off. In the case of deidentification, for instance, recall is arguably of more importance than precision, as argued in [35], where F_2-score was used to evaluate various deidentification systems.

V. CONCLUSION

We have presented a method for generating semantic features to be exploited by the learning algorithm when training a named entity recognition model; this is here evaluated in the context of deidentifying text-based health records. The method uses the small set of available annotations to learn a prototype vector for each named entity class by identifying the centroid of the annotated instances' representations in semantic space and using a cosine similarity threshold – set to maximize F_β-score – to determine whether or not a given word belongs to that class. As the semantic space is constructed over an unannotated corpus, the method is able to make effective use of large amounts of raw text data – often readily available – and may thereby reduce the human effort involved in creating annotated resources, which, to do for every problem and (sub-)domain, is prohibitively expensive.

When learning prototype vectors from the annotations, it was shown that by defining the centroid as the median, as opposed to the mean, value of each vector dimension, it was possible to separate the named entity classes in semantic space more purely. Generating semantic features with the use of these prototype vectors and providing them to the learning algorithm, in addition to a set of orthographic and syntactic features, leads to significant improvements on the deidentification task over a model without access to the semantic features. Moreover, generating features by combining semantic spaces constructed with different models of distributional semantics leads to further improvements.

ACKNOWLEDGMENT

This work was partly supported by the project High-Performance Data Mining for Drug Effect Detection at Stockholm University, funded by the Swedish Foundation for Strategic Research under grant IIS11-0053. The authors would also

like to thank Martin Duneld, Maria Skeppstedt and Jing Zhao for their valuable comments.

REFERENCES

[1] S. M. Meystre, G. K. Savova, K. C. Kipper-Schuler, and J. F. Hurdle, "Extracting information from textual documents in the electronic health record: a review of recent research," *Yearb Med Inform*, vol. 35, pp. 128–144, 2008.

[2] P. B. Jensen, L. J. Jensen, and S. Brunak, "Mining electronic health records: towards better research applications and clinical care," *Nature Reviews Genetics*, vol. 13, no. 6, pp. 395–405, 2012.

[3] C. Friedman, P. Kra, and A. Rzhetsky, "Two biomedical sublanguages: a description based on the theories of zellig harris," *Journal of biomedical informatics*, vol. 35, no. 4, pp. 222–235, 2002.

[4] S. Meystre and P. J. Haug, "Automation of a problem list using natural language processing," *BMC medical informatics and decision making*, vol. 5, no. 1, p. 30, 2005.

[5] F. S. Roque, P. B. Jensen, H. Schmock, M. Dalgaard, M. Andreatta, T. Hansen, K. Søeby, S. Bredkjær, A. Juul, T. Werge *et al.*, "Using electronic patient records to discover disease correlations and stratify patient cohorts," *PLoS computational biology*, vol. 7, no. 8, p. e1002141, 2011.

[6] W. W. Chapman, L. M. Christensen, M. M. Wagner, P. J. Haug, O. Ivanov, J. N. Dowling, and R. T. Olszewski, "Classifying free-text triage chief complaints into syndromic categories with natural language processing," *Artificial intelligence in medicine*, vol. 33, no. 1, pp. 31–40, 2005.

[7] R. Eriksson, P. B. Jensen, S. Frankild, L. J. Jensen, and S. Brunak, "Dictionary construction and identification of possible adverse drug events in danish clinical narrative text," *Journal of the American Medical Informatics Association*, vol. 20, no. 5, pp. 947–953, 2013.

[8] P. LePendu, S. V. Iyer, A. Bauer-Mehren, R. Harpaz, J. M. Mortensen, T. Podchiyska, T. A. Ferris, and N. H. Shah, "Pharmacovigilance using clinical notes," *Clinical pharmacology & therapeutics*, vol. 93, no. 6, pp. 547–555, 2013.

[9] S. Santiso, A. Pérez, K. Gojenola, I. Taldea, A. Casillas, and M. Oronoz, "Adverse drug event prediction combining shallow analysis and machine learning," in *Proceedings of the 5th International Workshop on Health Text Mining and Information Analysis (Louhi)*, 2014, pp. 85–89.

[10] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingvisticae Investigationes*, vol. 30, no. 1, pp. 3–26, 2007.

[11] J. L. Fernández-Alemán, I. C. Señor, P. Á. O. Lozoya, and A. Toval, "Security and privacy in electronic health records: A systematic literature review," *Journal of Biomedical Informatics*, vol. 46, no. 3, pp. 541–562, 2013.

[12] HIPAA Health Insurance Portability and Accountability (HIPAA), "U.S. Department of Health and Human Services, http://www.cdc.gov/mmwr/preview/mmwrhtml/m2e411a1.htm," 2003.

[13] S. M. Meystre, F. J. Friedlin, B. R. South, S. Shen, and M. H. Samore, "Automatic de-identification of textual documents in the electronic health record: a review of recent research," *BMC medical research methodology*, vol. 10, no. 1, p. 70, 2010.

[14] Ö. Uzuner, Y. Luo, and P. Szolovits, "Evaluating the state-of-the-art in automatic de-identification," *Journal of the American Medical Informatics Association*, vol. 14, no. 5, pp. 550–563, 2007.

[15] O. Ferrández, B. R. South, S. Shen, F. J. Friedlin, M. H. Samore, and S. M. Meystre, "BoB, a best-of-breed automated text de-identification system for VHA clinical documents," *Journal of the American Medical Informatics Association*, vol. 20, no. 1, pp. 77–83, 2013.

[16] S. Miller, J. Guinness, and A. Zamanian, "Name tagging with word clusters and discriminative training." in *HLT-NAACL*, vol. 4. Citeseer, 2004, pp. 337–342.

[17] J. Turian, L. Ratinov, and Y. Bengio, "Word representations: a simple and general method for semi-supervised learning," in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 384–394.

[18] T. Cohen and D. Widdows, "Empirical distributional semantics: methods and ical applications," *Journal of Biomedical Informatics*, vol. 42, no. 2, pp. 390–405, 2009.

[19] A. Henriksson, "Semantic Spaces of Clinical Text: Leveraging Distributional Semantics for Natural Language Processing of Electronic Health Records," Licentiate Thesis, Stockholm University, 2013.

[20] S. Jonnalagadda, T. Cohen, S. Wu, and G. Gonzalez, "Enhancing clinical concept extraction with distributional semantics," *Journal of biomedical informatics*, vol. 45, no. 1, pp. 129–140, 2012.

[21] P. Kanerva, J. Kristofersson, and A. Holst, "Random indexing of text samples for latent semantic analysis," in *Proceedings of the 22nd annual conference of the cognitive science society*, vol. 1036. Citeseer, 2000.

[22] A. Henriksson, H. Moen, M. Skeppstedt, V. Daudaravicius, and M. Duneld, "Synonym extraction and abbreviation expansion with ensembles of semantic spaces," *Journal of Biomedical Semantics*, vol. 5, p. 6, 2014.

[23] M. Skeppstedt, M. Ahltorp, and A. Henriksson, "Vocabulary expansion by semantic extraction of medical terms," in *Proceedings of the Symposium on Languages in Biology and Medicine (LBM)*, 2013.

[24] M. Sahlgren, A. Holst, and P. Kanerva, "Permutations as a means to encode order in word space," in *Proceedings of the 30th Annual Meeting of the Cognitive Science Society*, 2008, pp. 1300–1305.

[25] A. Henriksson, M. Conway, M. Duneld, and W. W. Chapman, "Identifying synonymy between SNOMED Clinical Terms of Varying Length Using Distributional Analysis of Electronic Health Records," in *AMIA Annual Symposium Proceedings*, 2013, pp. 600–609.

[26] H. Dalianis and H. Boström, "Releasing a Swedish clinical corpus after removing all words - de-identification experiments with conditional random fields and random forests," in *Proceedings of BioTxtM 2012*, 2012, pp. 45–48.

[27] C. Sutton and A. McCallum, "An introduction to conditional random fields for relational learning," *Introduction to statistical relational learning*, pp. 93–128, 2006.

[28] H. Dalianis, M. Hassel, A. Henriksson, and M. Skeppstedt, "Stockholm EPR Corpus: A Clinical Database Used to Improve Health Care," in *Swedish Language Technology Conference*, 2012.

[29] H. Dalianis, M. Hassel, and S. Velupillai, "The Stockholm EPR Corpus – Characteristics and some initial findings," in *Proceedings of the Symposium on Health Information Management Research*, 2009.

[30] H. Dalianis and S. Velupillai, "De-identifying Swedish clinical text-refinement of a gold standard and experiments with Conditional random fields," *J. Biomedical Semantics*, vol. 1, p. 6, 2010.

[31] S. Velupillai, H. Dalianis, M. Hassel, and G. H. Nilsson, "Developing a standard for de-identifying electronic patient records written in Swedish: Precision, recall and F-measure in a manual and computerized annotation trial," *International journal of medical informatics*, vol. 78, no. 12, pp. e19–e26, 2009.

[32] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[33] S. Garcia and F. Herrera, "An Extension on" Statistical Comparisons of Classifiers over Multiple Data Sets" for all Pairwise Comparisons," *Journal of Machine Learning Research*, vol. 9, no. 12, 2008.

[34] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural computation*, vol. 10, no. 7, pp. 1895–1923, 1998.

[35] O. Ferrández, B. R. South, S. Shen, F. J. Friedlin, M. H. Samore, and S. M. Meystre, "Evaluating current automatic de-identification methods with veteran's health administration clinical documents," *BMC medical research methodology*, vol. 12, no. 1, p. 109, 2012.